La Recherche Tabou

Par
Joseph Ayas
Marc André Viau

Plan de la présentation

- Introduction
- Explication détaillée de la Recherche Tabou (RT)
- Exemples
- Domaines d'application
- Ressources disponibles
- Conclusion

Définition de base de la RT

• **<u>Définition</u>** : méthode heuristique de recherche locale utilisée pour résoudre des problèmes complexes et/ou de très grande taille (souvent NP-durs). La RT a plusieurs applications en programmation non linéaire (PNL).

• **<u>Principe de base</u>** : poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré et ce,

⇒ en permettant des déplacements qui n'améliorent pas la solution

⇒ en utilisant le principe de <u>mémoire</u> pour éviter les retours en arrière (mouvements cycliques)

Définition de base de la RT

• **<u>Mémoire</u>** :

⇒ elle est représentée par une liste taboue qui contient les mouvements qui sont <u>temporairement</u> interdits

⇒ mouvements interdits ou solutions interdites

⇒ son rôle évolue au cours de la résolution: diversification (exploration de l'espace des solutions) vers intensification

• **<u>Exception aux interdictions</u>** : il est possible de violer une interdiction lorsqu'un mouvement interdit permet d'obtenir la meilleure solution enregistrée jusqu'à maintenant

Historique scientifique

Développement des heuristiques :

• Tendance dans les années 70 : techniques d'amélioration des solutions par recherche locale

⇒ procédure de recherche itérative qui améliore une solution de départ en lui appliquant une série de modifications locales (mouvements)

⇒ arrêt lorsqu'un optimum local est trouvé

• 1983 : une nouvelle heuristique apparaît, le *Recuit Simulé*

⇒ permet une exploration aléatoire contrôlée de l'espace des solutions

Historique scientifique

• 1986 : bien que son origine remonte à 1977, la RT n'est proposée qu'au milieu des années 80 par Fred Glover

⇒ méthode développée pour résoudre des problèmes combinatoires (la plupart NP-durs)

⇒ révolution de cette méthode par rapport aux autres: permet de surmonter le problème des optima locaux par l'utilisation de listes taboues (principe de mémoire)

• Par la suite : algorithmes génétiques, colonies de fourmis, …

Mise en contexte: « Fable des randonneurs »

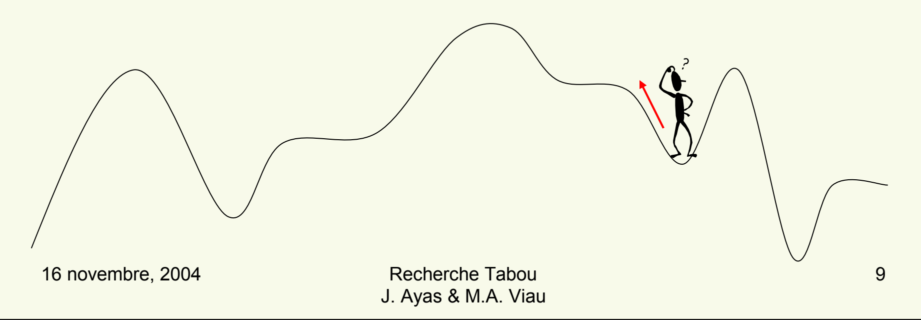
Un randonneur malchanceux , T. A. Bhoulx, est perdu dans une région montagneuse. Toutefois, il sait qu'une équipe de secours passe régulièrement par le point situé à la plus basse altitude dans la région. Ainsi, il doit se rendre à ce point pour attendre les secours. Comment s'y prendra-t-il ? Il ne connaît pas l'altitude de ce point et, à cause du brouillard, il ne voit pas autour de lui. Donc, arrivé à un croisement, il doit s'engager dans une direction pour voir si le chemin monte ou descend.

Mise en contexte: « Fable des randonneurs »

Tout d'abord, il commence par descendre tant qu'il peut, en choisissant le chemin de plus grande pente à chaque croisement.

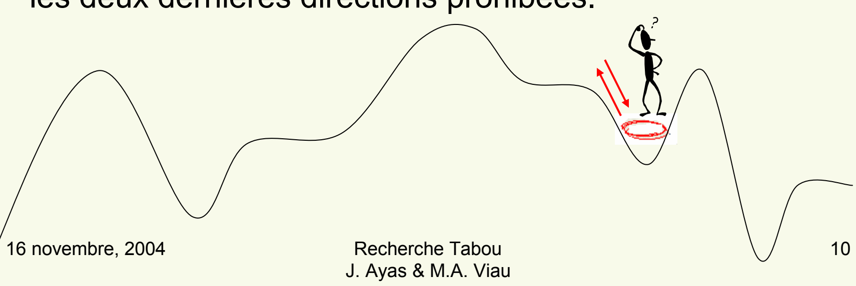
Mise en contexte: « Fable des randonneurs »

Puis, lorsqu'il n'y a plus de sentier menant vers le bas, il décide de suivre le chemin qui remonte avec la plus faible pente car il est conscient qu'il peut se trouver à un minimum local.



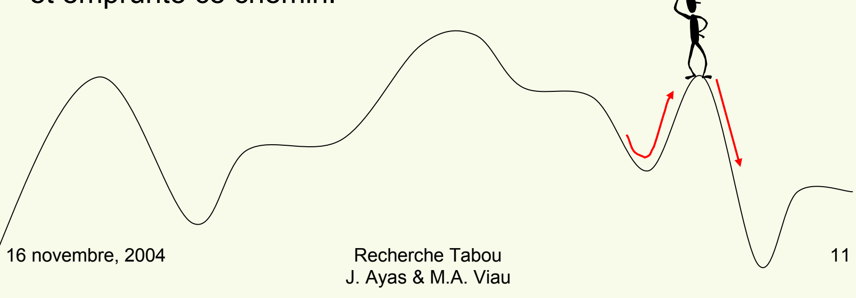
Mise en contexte: « Fable des randonneurs »

Toutefois, dès qu'il remonte, il redescend vers le point où il était. Cette stratégie ne fonctionne pas. Par conséquent, il décide de s'interdire de faire marche arrière en mémorisant la direction d'où il vient. Il est à noter que sa mémoire ne lui permet de mémoriser que les deux dernières directions prohibées.



Mise en contexte: « Fable des randonneurs »

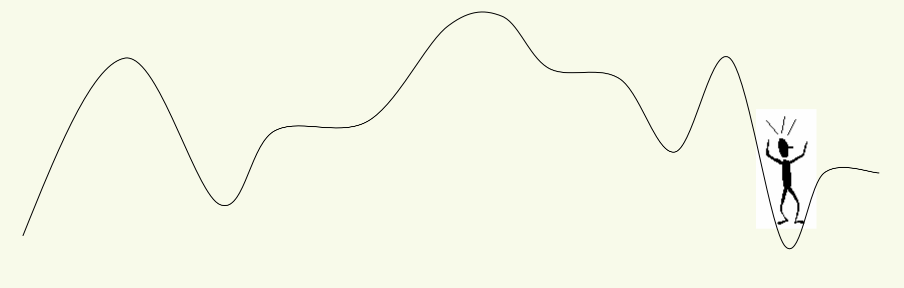
Cette nouvelle stratégie lui permet d'explorer des minimum locaux et d'en ressortir. À un moment donné, il arrive à un point où il décèle une forte pente descendante vers le sud. Toutefois, les directions mémorisées lui interdisent d'aller vers le sud car cette direction est prohibée. Il décide d'ignorer cette interdiction et emprunte ce chemin.



Introduction

Mise en contexte: « Fable des randonneurs »

Cette décision fut bénéfique: il arriva au point de plus basse altitude et attendit les secours qui ne tardèrent à arriver.



« Méta heuristique »

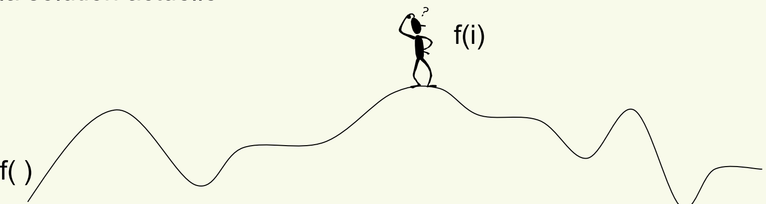
• notion de méta heuristique: souvent utilisée pour décrire la RT (≠ méthode exacte)

• une stratégie qui guide et modifie d'autres heuristiques afin de produire des solutions qui diffèrent de celles généralement obtenues dans la recherche d'un optimum local

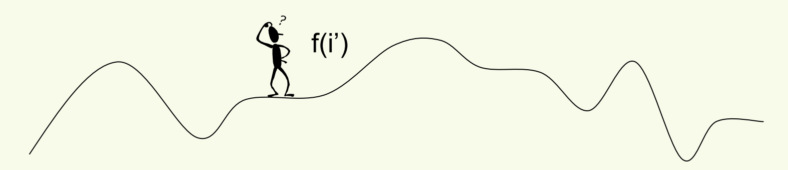
• ces heuristiques « guidées » peuvent se limiter à de simples descriptions et évaluations de déplacements permis pour passer d'une solution à une autre

Définition des variables

• i : la solution actuelle

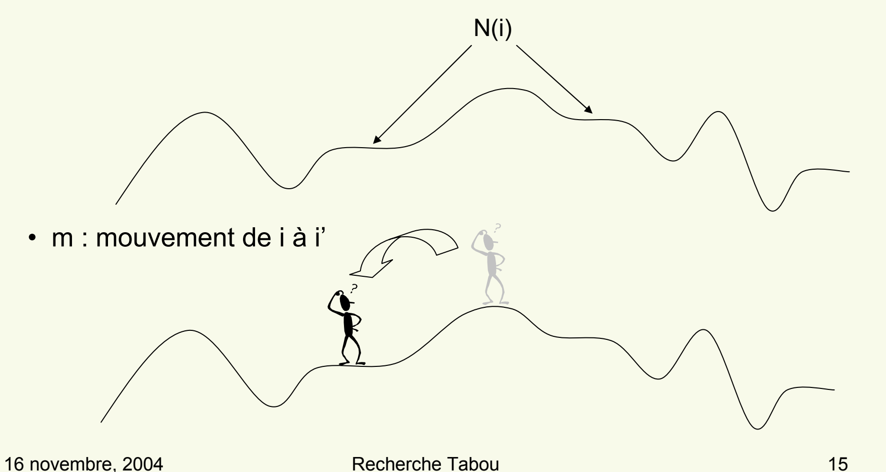


• i' : la prochaine solution atteinte (solution voisine)



Définition des variables

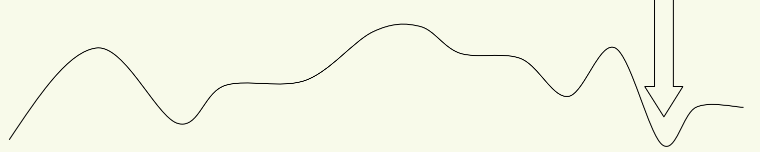
- N(i): l'espace de solutions voisines à i (l'ensemble des i')



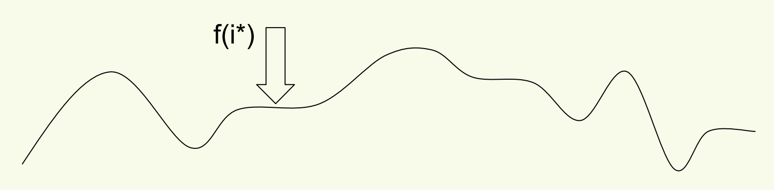
- m : mouvement de i à i'

Définition des variables

- $i_{globale}$: la solution optimale *globale* qui minimise la fonction objectif f( ).

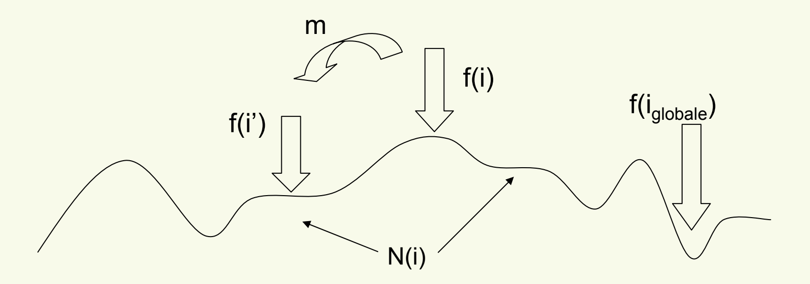


- i* : la solution optimale actuelle



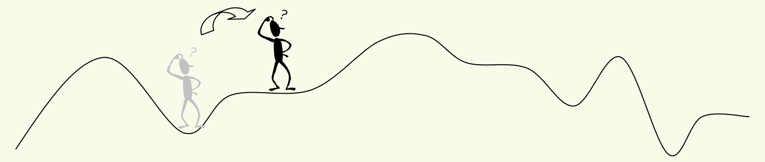
Résumé des variables

Et donc, jusqu'à présent, nous avons:



Définition des termes

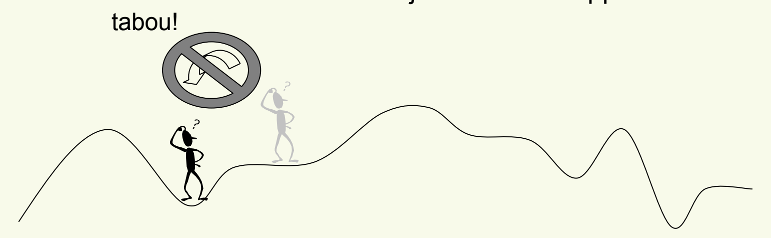
- Mouvement non améliorateur: un mouvement qui nous sortirait d'un minimum local i* en nous amenant à une solution voisine i' *pire* que l'actuelle.



La méthode tabou permet un mouvement non améliorateur, comme le permet le recuit simulé. La différence entre les 2 méthodes est que la RT choisira le meilleur i' dans N(i), l'ensemble des solutions voisines.

Définition des termes

- Mouvement tabou: un mouvement non souhaitable, comme si on redescendait à un minimum local d'où on vient juste de s'échapper.



Le mouvement est considéré tabou pour un nombre prédéterminé d'itérations. k représente l'index des itérations (l'itération actuelle).

Définition des termes

- T : liste des mouvements tabous. Il peut exister plusieurs listes simultanément. Les éléments de la liste sont t(i,m).

  Une liste T avec trop d'éléments peut devenir très restrictive. Il a été observé que trop de contraintes (tabous) forcent le programme à visiter des solutions voisines peu alléchantes à la prochaine itération.

  Une liste T contenant trop peu d'éléments peu s'avérer inutile et mener à des mouvements cycliques.

- a(i,m) : *critères d'aspiration*. Déterminent quand il est avantageux d'entreprendre m, malgré son statut tabou.

L’algorithme

Étape 1: choisir une solution initiale i dans S (l’ensemble des solutions)

Appliquer i* = i et k = 0

Étape 2: appliquer k = k+1 et générer un sous-ensemble de solutions en N(i,k) pour que:

- les mouvements tabous ne soient pas choisis
- un des critères d’aspiration a(i,m) soit applicable

Étape 3: choisir la meilleure solution i’ parmi l’ensemble de solutions voisines N(i,k)

Appliquer i = meilleur i’

L’algorithme (suite)

Étape 4: si f(i) <= f(i*), alors nous avons trouvé une meilleure solution

Appliquer i* = i

Étape 5: mettre à jour la liste T et les critères d’aspiration

Étape 6: si une condition d’arrêt est atteinte, stop.
Sinon, retour à Étape 2.

Condition d’arrêt: condition qui régira l’arrêt de l’algorithme.

ex: arrêt après 22 itérations (k = 22).

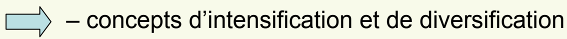
Améliorations

- La recherche de la solution optimale peut être améliorée. Voici quelques options:

  – choix stratégique de la solution initiale i. Ceci donnera une « bonne » valeur de f(i*)

  – *Intensifier* la recherche dans les voisinages de solutions qui semblent propices à mener à des solutions proches ou égales à l'optimum

  – *Diversifier* la recherche en éloignant celle-ci de voisinages peu propices à produire de bonne solutions



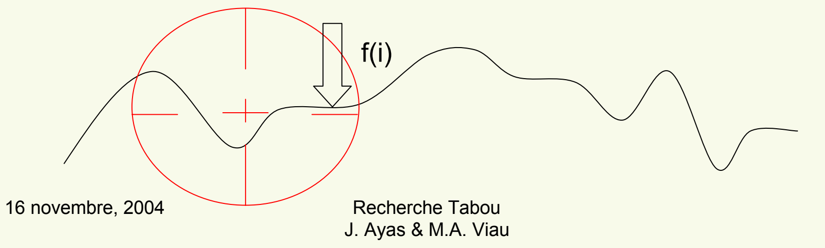
  – concepts d'intensification et de diversification

Intensification

La recherche est menée dans un voisinage N(i) de S, l'ensemble des solutions

Une haute priorité est donnée aux solutions f(i') qui ressemblent à la solution actuelle f(i)

Le résultat est donc une intensification de la recherche dans un certain secteur, dans un voisinage choisi:

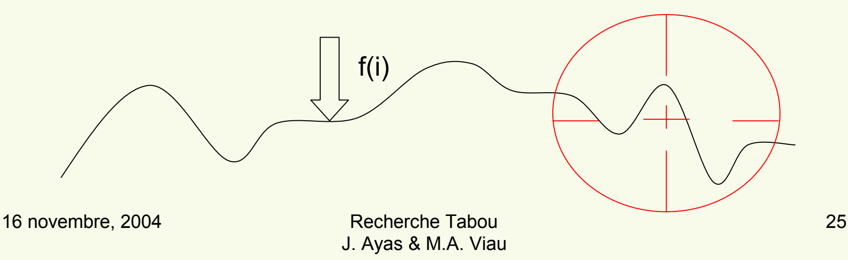


Diversification

La recherche est éloignée du voisinage N(i) actuelle de l'ensemble des solutions

Une haute priorité est donnée aux solutions f(i') d'une autre région que celle actuellement sous exploration

Le résultat : chercher ailleurs



Modifications à f( )

L'intensification et la diversification sont présentées comme des modifications à la fonction objectif.

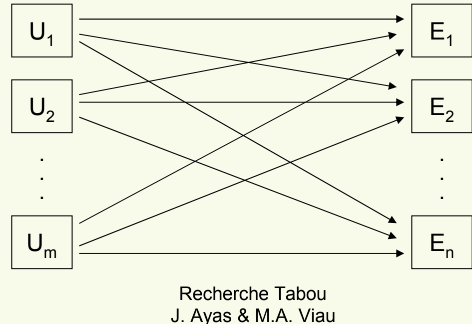
Pour *l'intensification*, une « pénalité » est attribuée à des solutions éloignées de l'actuelle. Ceci cause un gonflement de la fonction objectif : les solutions semblables seront donc privilégiées. Pour la *diversification*, l'effet est le contraire. Les solutions proches de l'actuelle sont pénalisées.

Donc: $\tilde{f}$ = f + intensification + diversification

Il est à noter que l'intensification et diversification se manifestent pendant quelques itérations k seulement, ensuite il y a alternance.

Exemple 1 – Transport

- La recherche tabou peut être utilisée pour le problème de transport que nous connaissons si bien:
- m usines
- n entrepôts
- Un coût fixe et un coût variable associés à l'utilisation d'une route

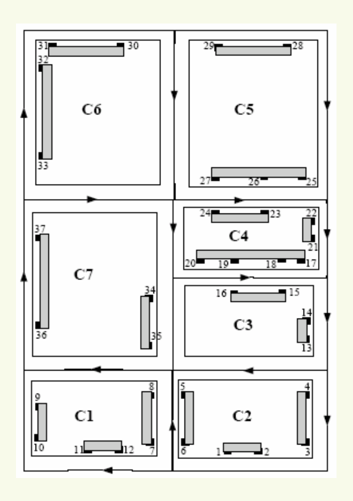


Exemple 1 – Transport

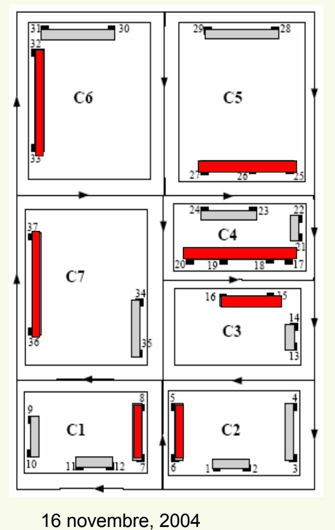
- Une recherche menée par Sun et al. en 1995 cherchait à démontrer les avantages de la RT dans le problème de transport.
- L'utilisation des attributs de mémoire et de visites récentes.
- Algorithme de descente: simplex.
- 15 problèmes de différentes tailles étudiés.
- Résultats comparés avec un algorithme de solution exacte.
- Résultats:
  - solution optimale trouvée: 12 en 15
  - 3 autres, la solution < 0.06% de l'optimale
  - vitesse avec RT: 1.63s de processeur (moyenne)
  - vitesse sans RT: 5888s de processeur (moyenne)
  - RT beaucoup plus efficace pour des problèmes de plus grande envergure

Exemple 2 – VGA

- La recherche tabou peut être utilisée pour le problème de véhicule guidé automatisé dans les entrepôts et au sein des systèmes manufacturiers
- Trouver la solution optimale des portes d'entrées et de sorties
- Étude menée par Chiang et Kouvelis en 1994



Exemple 2 – VGA

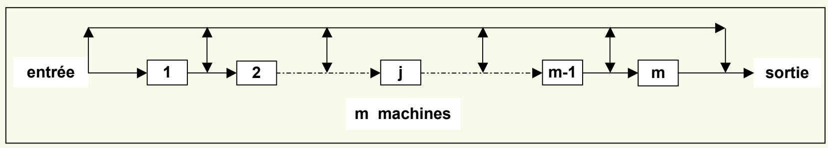


- 45 cas de VGA sous analyse
- Considération que les routes sont unidirectionnelles
- Algorithme utilisé met en valeur les avantages de la RT et du recuit simulé
- Résultats:
  - Domination de la RT comparée à d'autres heuristiques
  - différence de 0.855% moyenne des solutions optimales

Exemple 3 – Problème du *job shop*

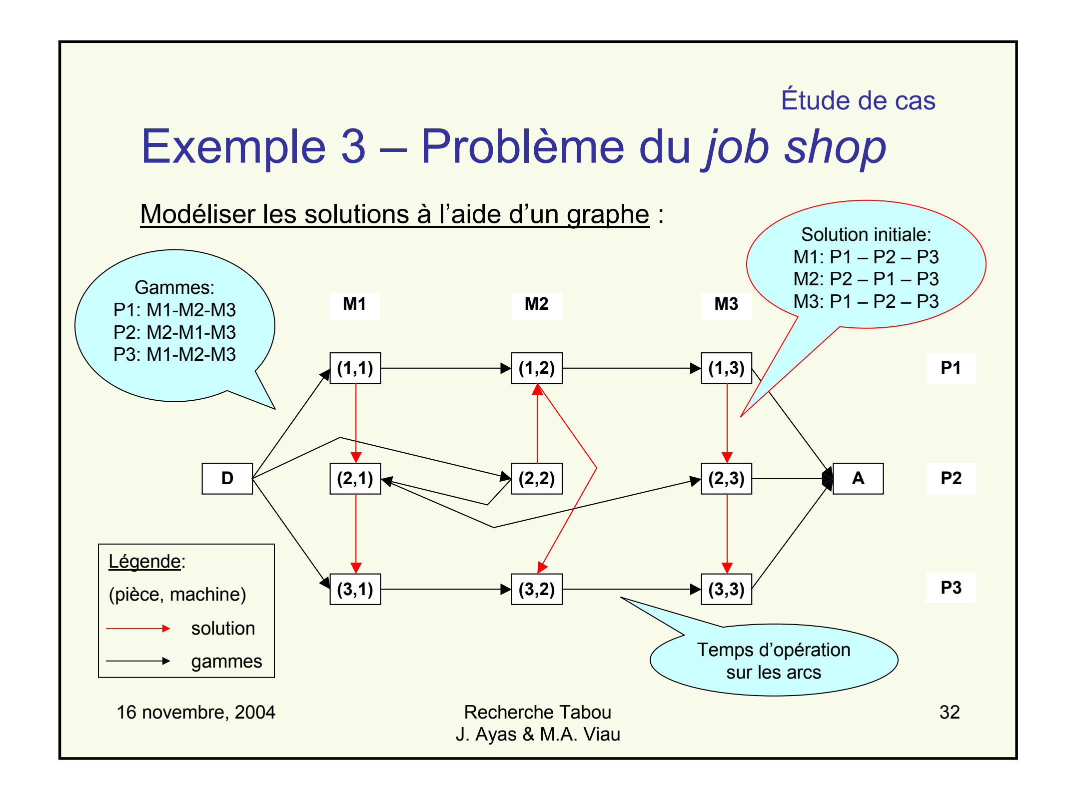
Les problèmes d'ordonnancement sur plusieurs machines (NP-complets) sont divisés en deux grandes classes:

- Problème de ***flow shop*** :
  - Exemple : ligne d'assemblage



- Problème de ***job shop*** :
  - les pièces n'ont pas la même progression sur les machines
  - n pièces, m machines ⇒ $(n!)^m$ cédules possibles
  - 10 pièces, 8 machines = 3 X 10^{52} cédules possibles
  - solution : utilisation d'heuristiques comme la RT

Exemple 3 – Problème du *job shop*



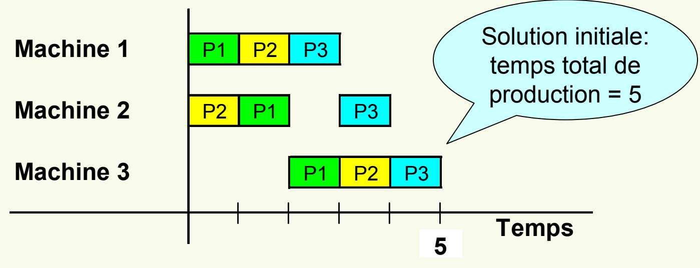
Exemple 3 – Problème du *job shop*

Déterminer la valeur de la fonction objectif :

- Fonction objectif = minimiser le temps total de production :

  ⇒ Minimiser le plus long chemin dans le graphe

- Diagramme de Gantt (supposons que toutes les opérations ont une durée unitaire) :



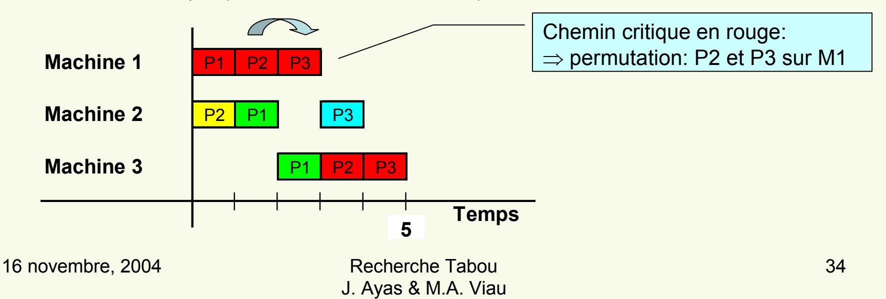
Exemple 3 – Problème du *job shop*

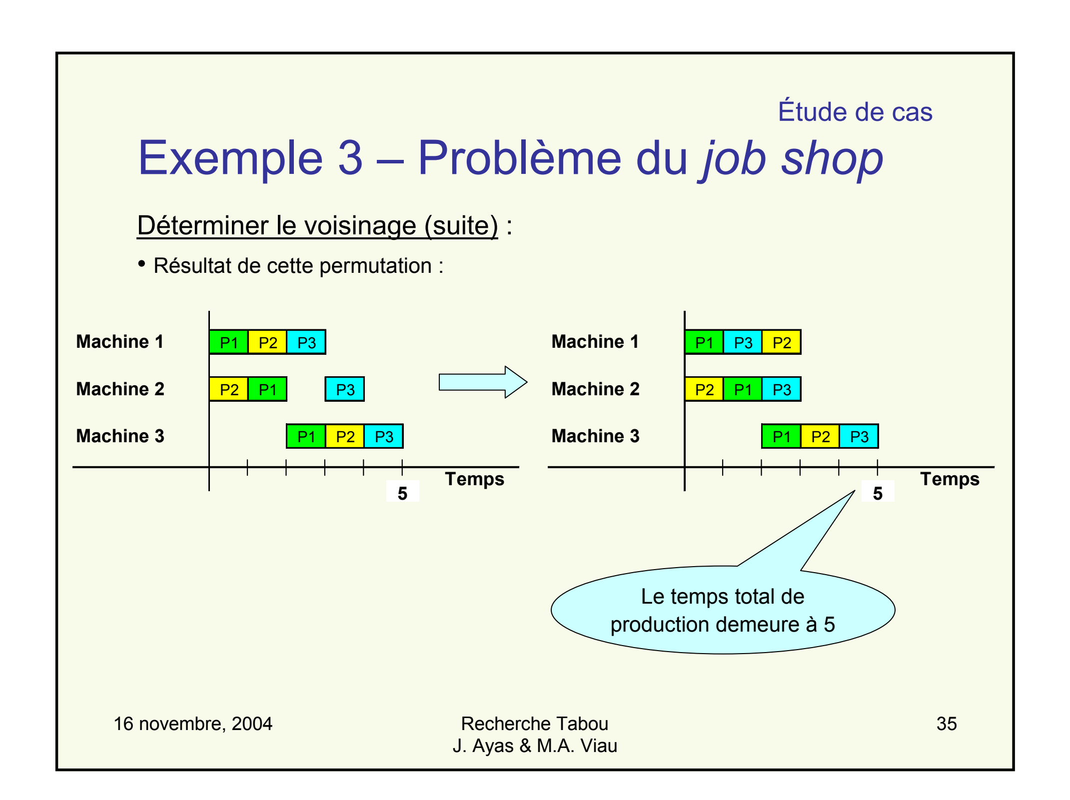
Déterminer le voisinage :

- Utilisation de la RT pour trouver une bonne solution puisqu'il est impossible de tester le temps total de production pour toutes les cédules applicables

- Comment déterminer le voisinage d'une solution ?

  ⇒ À partir d'une solution, on peut obtenir une solution voisine réalisable en permutant deux tâches consécutives sur le chemin critique (et sur une même machine)





Exemple 3 – Problème du *job shop*

Rôle de la RT :

- En fonction des solutions retenues et des permutations réalisées avec la RT, il est possible de modifier le graphe qui modélise le problème du *job shop*.

- Présentation de la résolution d'un problème de *job shop* dans *Excel* à l'aide de la RT

Étude de cas

Exemple 3 – Problème du *job shop*

Définition du problème :

- Ordonnancement pour un problème de *job shop*
- n = 10 pièces, m = 10 machines
- Hypothèse : chaque pièce doit passer 1 fois sur chaque machine
- Données :

Gammes opératoires

Pièces	Séquence sur les machines									
P1	1	2	3	4	5	6	7	8	9	10
P2	1	3	5	10	4	2	7	6	8	9
P3	2	1	4	3	9	6	8	7	10	5
P4	2	3	1	5	7	9				
P5	3	1	2	6	4	5				
P6	3	2	6	4	9	10				
P7	2	1	4	3	7	6				
P8	3	1	2	6	5	7				
P9	1	2	4	6	3	10				
P10	2	1	3	7	9	10				

Temps d'opérations

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
P1	29	78	9	36	49	11	62	56	44	21
P2	43	28	90	69	75	46	46	72	30	11
P3	85	91	74	39	33	10	89	12	90	45
P4	71	81	95	98	99	43	9	85	52	22
P5	6	22	14	26	69	61	53	49	21	72
P6	47	2	84	95	6	52	65	25	48	72
P7	37	46	13	61	55	21	32	30	89	32
P8	86	46	31	79	32	74	88	36	19	48
P9	76	69	85	76	26	51	40	89	74	11
P10	13	85	61	52	90	47	7	45	64	76

Étude de cas

Exemple 3 – Problème du *job shop*

Résolution à l'aide d'Excel (langage VBA):

1. Codage du graphe de base
2. Codage du calcul du plus long chemin pour la solution courante
3. Codage de l'algorithme permettant de déterminer les tâches critiques
4. Codage de l'algorithme permettant de déterminer les permutations possibles (voisinage)
5. Codage de l'algorithme permettant de modifier le graphe en fonction des permutations effectuées
6. Codage de l'**algorithme de RT**

Exemple 3 – Problème du *job shop*

Codage de l'algorithme de RT :

1- Créer une solution initiale, qui devient la solution courante (i) :

Solution	Séquence des pièces sur les machines									
M1	1	2	3	4	5	6	7	8	9	10
M2	1	2	3	4	5	6	7	8	9	10
M3	1	2	3	4	5	6	7	8	9	10
M4	1	2	3	4	5	6	7	8	9	10
M5	1	2	3	4	5	6	7	8	9	10
M6	1	2	3	4	5	6	7	8	9	10
M7	1	2	3	4	5	6	7	8	9	10
M8	1	2	3	4	5	6	7	8	9	10
M9	1	2	3	4	5	6	7	8	9	10
M10	1	2	3	4	5	6	7	8	9	10

2- Poser : i* (solution optimale) = i, f(i*) = f(i) = 3425

Exemple 3 – Problème du *job shop*

Codage de l'algorithme de RT (suite) :

3- Initialiser la matrice Tabou à 0

(la matrice Tabou contient toutes les permutations qui sont taboues: une permutation est taboue pendant 10 itérations)

4- Faire :

- 4.1- Pour la solution courante (i), déterminer toutes les permutations possibles (voisinage): ensemble N(i)
- 4.2- Initialiser la variable *meilleure solution voisine* à 100000
- 4.3- Déterminer la meilleure solution voisine à i. Pour chaque permutation dans N(i), faire :
  - 4.3.1- Permuter les tâches correspondantes dans le graphe courant i, pour obtenir i', et calculer le temps de production f(i')

Exemple 3 – Problème du *job shop*

Codage de l’algorithme de RT (suite) :

4.3.2- Si f(i’) est inférieur à la valeur de la variable *meilleure solution voisine* et que la permutation n’est pas tabou ou que f(i’) ′ f(i*), on retient la permutation en cours et on ajuste la variable *meilleure solution voisine*

4.4- Modifier la solution courante (i) avec la meilleure permutation trouvée (meilleure solution voisine)

4.5- Ajouter cette permutation dans la matrice Tabou (elle sera taboue pour les 10 prochaines itérations)

4.6- Si f(i)′ f(i*), ajuster i* et f(i*) en fonction de i

4.7- Tant qu’il ne se passe pas 100 itérations consécutives sans amélioration de i*, revenir à 4.1

Exemple 3 – Problème du *job shop*

Exemple de matrice Tabou (mémoire à court terme) :

Voici les 5 premières permutation effectuées:

P1	P2	M	Itération
8	9	4	11
9	8	4	11
9	10	9	12
10	9	9	12
1	2	10	13
2	1	10	13
3	4	5	14
4	3	5	14
2	3	2	15
3	2	2	15

La première permutation effectuée a été de permuter les pièces 8 et 9 sur la machine 4. Donc, les permutations 8 et 9 ainsi que 9 et 8 seront taboues jusqu’à l’itération 11.

Exemple 3 – Problème du *job shop*

Résultats de la RT :

- Temps de production initial: f(i*) = 3425
- Temps de production après 74 itérations: f(i*) = 1298
- Temps de résolution : court si l'algorithme est bien codé*
- Solution :

Machines	**Séquences sur les machines**									
M1	1	2	3	5	7	4	8	9	10	6
M2	3	1	4	7	2	5	6	8	10	9
M3	2	1	5	4	3	8	6	7	10	9
M4	1	3	2	7	5	6	9	4	10	8
M5	1	2	4	5	3	8	6	7	10	9
M6	1	2	3	5	6	8	7	9	10	4
M7	1	2	3	4	7	8	10	6	5	9
M8	1	3	2	4	5	6	7	9	8	10
M9	1	3	2	4	5	6	10	7	8	9
M10	2	1	3	7	6	5	10	4	9	8

Domaines concernés…

- Problèmes de transport
  - ⇒ confection de tournées de véhicules (contraintes de capacité, fenêtres de temps, transport multi-modes, multi-produits,…)
  - ⇒ ordonnancement de convois
  - ⇒ design de réseaux
  - ⇒ problème du voyageur de commerce
- Planification et ordonnancement
  - ⇒ *flow shop*
  - ⇒ ***job shop***
  - ⇒ fabrication en cellule
  - ⇒ planification de la main-d'œuvre requise

Domaines concernés…

- Optimisation de la production et gestion des inventaires
  - ⇒ production juste-à-temps
  - ⇒ planification d'inventaires multi-produits
  - ⇒ gestion des économies d'échelle
- Problèmes d'affectation et de localisation
- Optimisation de graphes
  - ⇒ coloration de graphes (Hertz, Taillard, De Werra)
  - ⇒ clique maximale (Gendreau, Soriano)

Domaines concernés…

- Télécommunications
  - ⇒ conception de réseaux (optiques, de service, …)
  - ⇒ routage d'appels
- Logique et intelligence artificielle
  - ⇒ reconnaissance et classification de formes
  - ⇒ réseaux de neurones
- Application à diverses technologies
  - ⇒ construction de stations spatiales
  - ⇒ distribution de puissance électrique
  - ⇒ inversion sismique

Domaines concernés…

- Création d'horaires
- Optimisation de structures
  - ⇒ structures des protéines
  - ⇒ séquençage d'ADN
- Techniques spécialisées
  - ⇒ *reactive Tabu search*
- Design
- Résolution en parallèle (*parallel computing*)
- Optimisation continue et stochastique
- Analyses financières

Tutoriaux – base de la RT

- GENDREAU, Michel. 2002. « An Introduction to Tabu Search », Centre de recherche sur les transports & Département d'informatique et de recherche opérationnelle – Université de Montréal. [En ligne]. http://www.ifi.uio.no/infheur/Bakgrunn/Intro_to_TS_Gendreau.htm
- HERTZ, Alain, Éric TAILLARD & Dominique DE WERRA. 1997. « A Tutorial on Tabu Search », dans le livre *Local search in combinatorial optimization*, p.121-136.
- GLOVER, Fred. 1990. « Tabu Search: A Tutorial », *Special Issue on the Practice of Mathematical Programming*, Interfaces, Vol. 20, No.1, p.74-94.

Groupes de recherche

- *Hearin Center for Enterprise Science* : http://hces.bus.olemiss.edu/

  ⇒ Collaboration inter-universitaire pour le développement de la recherche opérationnelle et de la science de la gestion (États-Unis)

  ⇒ Application de la RO pour des problèmes industriels

  ⇒ Recherche fondamentale

- Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle (IRIDIA): http://iridia.ulb.ac.be/

  ⇒ Recherche fondamentale : méta-heuristiques, modélisation de systèmes complexes, robotique autonome, *data mining*, …

- Institut d'informatique appliquée (INA) : http://ina.eivd.ch/newina/

  ⇒ Gestion de l'information, systèmes d'aide à la décision, …

Groupes de recherche

- Laboratoire de recherche en informatique PRiSM : http://www.prism.uvsq.fr/recherche/themes/aoc/opale/

  ⇒ Équipe de recherche OPALE : optimisation parallèle

Chez nous…

- Groupe d'études et de recherche en analyse des décisions (GERAD): http://www.gerad.ca/

- Centre de recherche sur les transports (CRT): www.crt.umontreal.ca/

Conférences

- *Metaheuristics International Conference* : http://www.mic2005.org/
  ⇒ méthodes basées sur la recherche locale (dont la RT)
  ⇒ application des méta-heuristiques
  ⇒ recherche fondamentale

- *Adaptive Memory and Evolution: Tabu Search and Scatter Search*
  http://hces.bus.olemiss.edu/events/schedule.html
  ⇒ conférence réservée à la recherche Tabou et à la recherche dispersée
  ⇒ présentée à l'Université du Mississippi en 2001

- *Operations Research International Conference :*
  http://center.uvt.nl/congres/
  ⇒ conférence annuelle couvrant tous les domaines de la RO

Sites web

- Site de Fred Glover : http://www.tabusearch.net
  ⇒ informations de base sur la RT
  ⇒ exemples d'application
  ⇒ ressources disponibles

- Coloration de graphes:
  http://membres.lycos.fr/dthiery/tabou/main.htm
  ⇒ introduction à la RT
  ⇒ exemple de coloration [en ligne]

- *OpenTS* : http://www.coin-or.org/OpenTS/index.html
  ⇒ modèle d'application java permettant de faciliter l'implantation de la RT
  ⇒ tutoriel, manuel d'utilisateur, aide en ligne, …

Livres

- GLOVER, Fred et Manuel LAGUNA. 1997. *Tabu Search*, Boston : Kluwer Academic Publishers, 382p.
  ⇒ référence dans le domaine
  ⇒ écrit par Fred Glover, qui est à l'origine de la RT
  ⇒ montre les concepts avancés de la méthode
  ⇒ énumère de nombreuses applications possibles

- REEVES, Colin. 1993. *Modern heuristic techniques for combinatorial problems,* New York: John Wiley & Sons, 320p.
  ⇒ contient un chapitre sur la RT

Revues

- ***Computers & Operations Research***
- ***European Journal of Operational Research***
- ***Applied Mathematics and Computation***
- ***Journal of Heuristics***
- *Computers & Industrial Engineering*
- *Mathematical and Computer Modelling*
- *Journal of Intelligent Manufacturing*
- *Engineering Applications of Artificial Intelligence*
- *Journal of Production Research*

Logiciels commerciaux

- Optquest
  - Crystal Ball
  - AnyLogic
  - Arena
  - FlexSim
  - Etc.

  Les produits Optquest intègrent la RT à même leur engin d'optimisation

- GAMS/DICOPT (BARON complement)
- HEURO (Integrated Optimization Environment)
- LGO (Lipschitz Global Optimization)

Logiciels commerciaux

- Building Reusable Software Components for Heuristic Search: création de composantes pour la RT

  utilisant C++ et HotFrame (Heuristic OpTimization FRAMEwork)

- Excel, Visual Basic (tel le problème –» exemple 3)

- Langages de programmation (ex: C, C++, Java etc.)

- AMPL (langage de modélisation)

Logiciels projets universitaires



Proyecto MALLBA: Code en C pour recherche tabou

```
bool TabuStorage::is_in_tabu_storage (const Movement& mov, const Solution& sol,
            const State& state) const
            {list_item it;
            bool found = false;
            int  i=0;
            while ((!found) && (i<tlist.size())) {                                   it
            tlist[i];
            found = (mov == tlist[it]);
            i++; }
            return found;
      }
```


Logiciels projets universitaires



Work orders raised in
MFG/Pro
(an Enterprise Resource Planning system)

Exported to
The Planner
to create a schedule

Inputs for the scheduling package

Data file
(details of items, machine characteristics and performance measures)

Schedule downloaded from The Planner into
Excel

SET

Functions of SET

Schedule opened in
SET

Scheduling

Rule-based | Tabu Search | Manual changes

Schedule evaluated in terms of a variety of performance measures

Schedule saved

Projet couplant
MFG/Pro (ERP),
SET (scheduling and
evaluation tool set)
et Excel pour améliorer
l'efficacité des cédules de production d'imprimantes.

Conclusions

- Incorpore de la mémoire dans la stratégie de recherche
- Permet les mouvements non-améliorateurs
- Offre des économies de temps de résolution pour des programmes de grosse taille
- Ne génère pas toujours la solution optimale (heuristique)
- Une liste taboue trop longue peut être restrictive. Par contre, une liste trop courte risque de s’avérer inutile.

Conclusions

- La RT est hautement adaptable au problème sous étude:
  - Itérations
  - Listes taboues
  - Choix de départ stratégique
  - Intensification et diversification
- Richesse d’études, références et recherches sur la RT et ses applications
- C’est une méthode toujours sous étude et faisant l’objet d’améliorations continuelles.

Référence supplémentaire

HERTZ, Alain. *IND4403 – Méthodes quantitatives en productique*, notes de cours, École Polytechnique de Montréal, Département de mathématiques et de génie industriel, Hiver 2004.

Riopel D., Langevin A. Savard G., *Flow Path Design for an Automated Guided Vehicle System*, Les Cahiers du GERAD G-98-32, juillet 1998.

Références supplémentaires: Internet – logiciels

Produits OptQuest
http://www.opttek.com/products/index.html

GAMS/DICOPT, HEURO, LGO
http://www.cas.mcmaster.ca/~cs777/presentations/GlobOpt_Soft/267,12,Commercial Systems

Proyecto MALLBA (Universitat Politécnica de Catalunya)
http://www.lsi.upc.es/~mallba/public/library/TabuSearch/#contacts

SET (University of Canterbury)
http://www.mang.canterbury.ac.nz/courseinfo/msci/msci480/Noonan_Dodgson/

Building Reusable Software Components for Heuristic Search
http://www.rrz.uni-hamburg.de/IWI/fink/fvw_or98.pdf